



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/683,929	10/09/2003	John W. Rapp	1934-13-3	2222
7590 Bryan A. Santarelli GRAYBEAL JACKSON HALEY LLP Suite 350 155 - 108th Avenue NE Bellevue, WA 98004-5901				
			EXAMINER HUISMAN, DAVID J	
			ART UNIT 2183	PAPER NUMBER
			MAIL DATE 09/24/2010	DELIVERY MODE PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary

Application No.

10/683,929

Applicant(s)

RAPP ET AL.

Examiner

DAVID J. HUISMAN

Art Unit

2183

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 07 June 2010.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-16, 41-50 and 66-85 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☒ Claim(s) 66, 67, 80 and 81 is/are allowed.
- 6) ☒ Claim(s) 1-16, 41-50, 68-79 and 82-85 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 11 March 2004 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
- Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
- Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftperson's Patent Drawing Review (PTO-948)
- 3) ☒ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date 7/6/10
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date _____
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: _____

DETAILED ACTION

1. Claims 1-16, 41-50, and 66-85 have been examined.

Claim Objections

2. Claim 1 is objected to because of the following informalities: It would appear to be more correct if applicant replaced "instruction:" in line 4, with --instruction, to:--, and deleted "to" from the beginning of each subsequent step. Appropriate correction is required.
3. Claims 6-8, 66-68, 71-75, and 78-79 are objected for the same reason that claim 1 is objected to.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1-12, 15-16, 41-46, 49-50, 69-70, 73, 76-77, and 83 are rejected under 35 U.S.C. 103(a) as being unpatentable over Inagaki et al., U.S. Patent No. 7,177,310 (herein referred to as Inagaki) in view of the examiner's taking of Official Notice.
6. Referring to claim 1, Inagaki has taught a pipeline accelerator, comprising:
a) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".

b) a hardwired circuit coupled to the memory, including processing accelerators (see Fig.1, at least some of components 9 and 11-14), and operable to:

b1) receive a message that includes data and that includes a header having information indicating at least one but fewer than all of the processing accelerators by receiving the data and the information on at least one common bus line. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Packet 134 of Fig.3 is received on the upper bus 1. The destination 163 in the header indicates which of the accelerators will be used to process the message data.

b2) extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with the indicated at least one processing accelerator without first using a virtual address corresponding to the indicated at least one processing accelerator. See Fig.1 and column 6, lines 29-35, and note that the data to be processed is stored in the appropriate buffer of buffers 150-154. The data is then retrieved from the buffer and processed by the corresponding accelerator of accelerators 9 and 11-14. It should be noted that, although a virtual address is used to indicate a pipeline, the use of this virtual address occurs before processing. The processing itself does not make use of the virtual address. It merely carries out the appropriate operation on the packet data. Hence, it can be said that the data is processed without first using a virtual address. The virtual address is used prior to processing.

b3) provide the processed data to an external source. See Fig.1, column 6, lines 61-65, and column 8, lines 62-67. Note that after the data is processed, it is routed to an external location.

b4) Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

b5) Inagaki has also not taught that the processing accelerators are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki's processing accelerators to be pipelined.

7. Referring to claim 2, Inagaki, as modified, has taught that pipeline accelerator of claim 1. Inagaki has not explicitly taught that the memory is disposed on a first integrated circuit and the

pipeline circuit is disposed on a second integrated circuit. However, the separation of components into distinct integrated circuits is not a patentable feature. One of ordinary skill in the art would have recognized that separation of components onto separate chips allows for increased flexibility and reduced repair cost because if just a portion of a system is to be upgraded or repaired, then only that integrated circuit would be upgraded or repaired, as opposed to upgrading or repairing the whole system (if it were on a single chip). As a result, it would have been obvious to modify Inagaki such that the memory 150-154 is implemented on a separate integrated circuit from that of the pipeline circuit 9 and 11-14. Also, see Nerwin v. Erlichman 168 USPQ 177 (1969).

8. Referring to claim 3, Inagaki, as modified, has taught the pipeline accelerator of claim 1. Inagaki has not taught that the pipeline circuit is disposed on a field-programmable gate array. However, Official Notice is taken that a field-programmable gate arrays (FPGA) and its advantages are well known and accepted in the art. Specifically, an FPGA may be efficiently reprogrammed by a designer after manufacture, thereby realizing specialized circuitry for processing in particular environments. The reprogrammability also allows a designer to upgrade, completely change, or otherwise modify a configuration as needed for increased efficiency. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the pipeline circuit is disposed on a field-programmable gate array.
9. Referring to claim 4, Inagaki, as modified, has taught the pipeline accelerator of claim 1 wherein the pipeline circuit is operable to provide the processed data to the external source by: loading the processed data into the memory; retrieving the processed data from the memory; and providing the retrieved processed data to the external source. See Fig.1 and note that after

processing, the processed data is loaded into and retrieved from memory components 119 and 124 and then provided to the external source.

10. Referring to claim 5, Inagaki, as modified, has taught the pipeline accelerator of claim 1, wherein the external source comprises a processor; and the pipeline circuit is operable to receive the data from the processor. Inagaki is concerned with transferring and processing data in a network. Hence, processors are sources and destinations of data.

11. Referring to claim 6, Inagaki has taught a computing machine, comprising:

a) a processor operable to broadcast a message that includes data and that includes a header having information identifying at least one but fewer than all destination accelerators of the data. See Fig.1, and note destination accelerators 9 and 11-14. Also, see column 5, line 65, to column 6, line 17. Packet 134 of Fig.3 is received on the upper bus 1 (inherently from some processor broadcasting an IP packet. Also, component 30 in Fig.1 may be considered a processor). The destination 163 in the header indicates which of the accelerators will be used to process the message data.

b) an accelerator coupled to the processor and comprising:

b1) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".

b2) a hardwired circuit coupled to the memory, including at least one processing component (see Fig.1), and operable to:

- receive the message from the processor by receiving the data and the information via at least one same bus line. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Packet 134 of Fig.3 is received on the upper bus 1. The destination

163 in the header indicates which of the accelerators will be used to process the message data.

- extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with the identified at least one destination accelerator without first referencing a virtual address corresponding to the identified at least one destination accelerator. See Fig.1 and column 6, lines 29-35, and note that the data to be processed is stored in the appropriate buffer of buffers 150-154. The data is then retrieved from the buffer and processed by the corresponding accelerator of accelerators 9 and 11-14. It should be noted that, although a virtual address is referenced to indicate a pipeline, the referencing of this virtual address occurs before processing. The processing itself does not reference the virtual address. It merely carries out the appropriate operation on the packet data. Hence, it can be said that the data is processed without first referencing a virtual address. The virtual address is referenced prior to processing to select a pipeline for processing.
- provide the processed data to the processor. See Fig.1, column 6, lines 61-65, and column 8, lines 62-67. Note that after the data is processed, it is routed back to the processor 30.
- Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to

perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

- Inagaki has also not taught that at least some of the above components are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify at least some of Inagaki's components to be pipelined.

12. Referring to claim 7, Inagaki has taught an accelerator, comprising:

- a) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".
- b) a hardwired circuit (Fig.1) coupled to the memory, and operable to:

b1) receive data without receiving, with the data, information corresponding to a post-processing destination of the data. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Essentially, a packet is received on the upper bus 1 and ultimately processed. The post-processing destination isn't added to the packet until after it is processed. See column 6, line 61, to column 7, line 32.

b2) process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.1 and note that the data is processed by at least one of processing units 9 and 11-14 before being stored in and ultimately retrieved from memory 119/124.

b3) generate a message header that includes first information indicating a destination of the data, generate a message that includes the processed data and the header, and provide the message to an external source. See Fig.1, column 7, lines 14-32, and column 8, lines 62-67.

b4) Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the

claimed steps without executing a program instruction, but instead in hardware alone.

For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

b5) Inagaki has also not taught at least some pipelined components. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki's accelerator and circuit to be pipelined.

13. Referring to claim 8, Inagaki has taught a computing machine comprising:

a) a processor operable to run at least one software application. See either Fig.2, component 2, or Fig.1, component 30, both of which inherently execute applications (i.e., perform some function). Note that both may collectively be referred to as a processor as well.

b) an accelerator coupled to the processor (see Fig.1) and comprising:

b1) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".

b2) a hardwired circuit (Fig.1) coupled to the memory, and operable to:

- receive data from the processor without receiving, with the data, information corresponding to a post-processing destination of the data. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Essentially, a packet is received

on the upper bus 1 and ultimately processed. The post-processing destination isn't added to the packet until after it is processed. See column 6, line 61, to column 7, line 32.

- process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.1 and note that the data is processed by at least one of processing units 9 and 11-14 before being stored in and ultimately retrieved from memory 119/124.
- generate a message header that includes, for the processed data, information that indicates a destination software application running on the processor, generate a message that includes the retrieved processed data and the message header, and provide the message to the processor. See Fig.1, column 7, lines 14-32, and column 8, lines 62-67. The destination is inherently to a processor in a network, and hence to an application running on that processor. Also, the packet's destination is Fig.1, component 30, which is part of the collective processor.
- Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa.

The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

- Inagaki has also not taught that at least some of the above components are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify at least some of Inagaki's components to be pipelined.

14. Referring to claim 9, Inagaki has taught a pipeline accelerator comprising:

a) first and second memories. See 1, and note first memory 150-154 and 117 and second memory 119 and 124.

b) a hardwired circuit (Fig.1) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having first information specifying at

least one destination hardwired processing element, to extract the raw data from the message, and to load the raw data into the first memory without first retrieving a virtual address corresponding to the at least one destination hardwired processing element specified by the first information. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Packet 134 of Fig.3 is received on the upper bus 1. The destination 163 in the header indicates which of the accelerators will be used to process the message data. Note also that the data is loaded into the first memory before processing (see Fig.1). It should be noted that, although a virtual address is retrieved to indicate a pipeline, the retrieval of this virtual address occurs before processing. The processing itself does not retrieve the virtual address. It merely carries out the appropriate operation on the packet data. Hence, it can be said that the data is processed without first retrieving a virtual address. The virtual address is retrieved prior to processing in order to select a pipeline for processing.

b2) hardwired processing elements including the specified at least one destination hardwired processing element and including at least one other processing element, the specified at least one destination hardwired processing element operable to process data. See Fig.1, component 9 and 11-14.

b3) an interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the at least one destination hardwired pipeline specified by the first information, and load processed data from the hardwired pipeline into the second memory. See Fig.1 and column 6, line 61, to column 7, line 13.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having second information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source by providing the processed data and the second information to the external source via at least one same bus line. See Fig.1, column 7, lines 14-32, and column 8, lines 62-67.

b5) Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

b6) Inagaki has also not taught that at least some of the above components are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different

items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify at least some of Inagaki's components to be pipelined.

15. Referring to claim 10, Inagaki, as modified, has taught a pipeline accelerator as described in claim 9 wherein the first and second memories each include respective first and second ports, the input-data handler is operable to load the raw data via the first port of the first memory, the pipeline interface is operable to retrieve the raw data via the second port of the first memory and to load the processed data via the first port of the second memory, and the output-data handler is operable to retrieve the processed data via the second port of the second memory. See Fig.1.

16. Referring to claim 11, Inagaki, as modified, has taught a pipeline accelerator as described in claim 9. Inagaki has not taught a third memory coupled to the hardwired-pipeline circuit, wherein the hardwired pipeline is operable to generate intermediate data while processing the raw data, and wherein the pipeline interface is operable to load the intermediate data into the third memory and to retrieve the intermediate data from the third memory. However, Official Notice is taken that using memories of some sort to process data is well known and accepted in the art. For instance, in a CPU (Fig.1, component 9), memories such as a register file and cache are well known and advantageous for providing fast, temporary storage of data during processing. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to include a third memory as described above.

17. Referring to claim 12, Inagaki, as modified, has taught that pipeline accelerator of claim 9.

a) Inagaki has not taught that the pipeline circuit is disposed on a field-programmable gate array. However, Official Notice is taken that a field-programmable gate arrays (FPGA) and its advantages are well known and accepted in the art. Specifically, an FPGA may be efficiently reprogrammed by a designer after manufacture, thereby realizing specialized circuitry for processing in particular environments. The reprogrammability also allows a designer to upgrade, completely change, or otherwise modify a configuration as needed for increased efficiency. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the pipeline circuit is disposed on a field-programmable gate array.

b) Inagaki has further not taught that the first and second memories are respectively disposed on first and second integrated circuits. However, the separation of components into distinct integrated circuits is not a patentable feature. One of ordinary skill in the art would have recognized that separation of components onto separate chips allows for increased flexibility and reduced repair cost because if just a portion of a system is to be upgraded or repaired, then only that integrated circuit would be upgraded or repaired, as opposed to upgrading or repairing the whole system (if it were on a single chip). As a result, it would have been obvious to modify Inagaki such that that the first and second memories are respectively disposed on first and second integrated circuits. Also, see *Nerwin v. Erlichman* 168 USPQ 177 (1969).

18. Referring to claim 15, Inagaki, as modified, has taught a pipeline accelerator as described in claim 9.

a) Inagaki, as modified, has further taught that each of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler has a respective operating configuration. That is,

each component in Inagaki operates in a specific fashion and consequently, each component inherently has a respective operating configuration.

b) Inagaki, as modified, has further taught a configuration manager coupled to and operable to set the operating configurations of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler. That is, it is the inherent nature of an FPGA to be coupled to a configuration manager so that the FPGA may be programmed to include the desired functionality.

19. Referring to claim 16, Inagaki, as modified, has taught a pipeline accelerator as described in claim 9.

a) Inagaki, as modified, has inherently taught that each of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler has a respective operating status. That is, at the very least, each component in the system is either operating or not operating (and these are statuses).

b) Inagaki, as modified, has not taught an exception manager coupled to and operable to identify an exception in the input-data handler, hardwired pipeline, pipeline interface, or output-data handler in response to the operating statuses. However, Official Notice is taken that checking for errors in a pipeline during processing is well known and accepted in the art. Any type of error which causes an exception should be monitored so that the system can take appropriate action to correct the error. As a result, in order to ensure proper execution, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that an exception manager is implemented to detect exceptions.

20. Referring to claim 41, the method of claim 41 is performed by the circuit of claims 1 and 9. Consequently, claim 41 is rejected for the same reasons set forth in the rejection of claims 1 and 9. Also, though Inagaki has not taught that the header has information indicating a size of the message, the examiner asserts that this concept is well known and accepted in the art. By indicating the size of the message, it is clear how much data is to be received/processed. This is useful in at least systems that handle variable length packets. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to indicate a size of the message within the header.

21. Referring to claim 42, Inagaki, as modified, has taught a method as described in claim 41. Furthermore, the method of claim 42 is performed by the circuit of claim 4. Consequently, claim 42 is rejected for the same reasons set forth in the rejection of claim 4.

22. Referring to claim 43, the method of claim 43 is performed by the accelerator of claim 7. Consequently, claim 43 is rejected for the same reasons set forth in the rejection of claim 7. In addition, the providing the message to an external source comprises providing on a single bus. See Fig.1 and note that the processed data is outputted via a single bus.

23. Referring to claim 44, the method of claim 44 is performed by the circuit of claims 1 and 9. Consequently, claim 44 is rejected for the same reasons set forth in the rejection of claims 1 and 9.

24. Referring to claim 45, Inagaki, as modified, has taught a method as described in claim 44. Furthermore, the method of claim 45 is performed by the circuit of claim 10. Consequently, claim 45 is rejected for the same reasons set forth in the rejection of claim 10.

25. Referring to claim 46, Inagaki, as modified, has taught a method as described in claim 44. Furthermore, the method of claim 46 is performed by the circuit of claim 11. Consequently, claim 46 is rejected for the same reasons set forth in the rejection of claim 11.

26. Referring to claim 49, Inagaki, as modified, has taught a method as described in claim 44, further comprising setting parameters for loading and retrieving the raw data, processing the retrieved data, and loading and providing the processed data. Recall from Fig.1 that data is sent to buffers before/after processing. Hence, pointers parameters need to be set to load and store data from/to buffers/queues, etc.

27. Referring to claim 50, Inagaki, as modified, has taught a method as described in claim 44. While Inagaki has not explicitly taught determining whether an error occurs during the loading and retrieving of the raw data, the processing of the retrieved data, and the loading and providing of the processed data, Official Notice is taken that checking for errors during processing is well known and accepted in the art. Any type of error, which causes an exception, should be monitored so that the system can take appropriate action to correct the error. As a result, in order to ensure proper execution, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to determining whether an error occurs during the loading and retrieving of the raw data, the processing of the retrieved data, and the loading and providing of the processed data.

28. Referring to claim 69, Inagaki, as modified, has taught the pipeline accelerator as described in claim 7, wherein the hardwired-pipeline circuit is further operable to:

a) store in association with the processed data second information indicating the destination of the processed data. See Fig.3, component 130. The data and MAC address 160 (second

information) are stored together in memory (see column 6, lines 22-35)

b) generate the message header in response to the second information. See Fig.3 and column 6, lines 9-21. An Ethernet header is generated in response to the MAC address.

29. Referring to claim 70, Inagaki, as modified, has taught a pipeline accelerator as described in claim 69 wherein the second information equals the first information. See Fig.3 and column 6, lines 9-35. Basically, a destination IP (first information) is mapped to a particular MAC address (second information), which in turn identifies the pipeline to perform the processing. Because they are mapped 1-to-1, they are equal. That is, both the IP and MAC identify the pipeline.

30. Referring to claim 73, claim 73 is rejected for the same reasons set forth in the rejections of claims 9 and 66.

31. Referring to claim 76, claims 76 is rejected for reasons set forth in rejections of previous independent claims (most notably, but not limited to, 9 and 73). Also, Inagaki has taught wherein the pipeline interface is further operable, without executing a program instruction, to store in association with the processed data second information indicating the destination of the processed data, and wherein the output-data handler is further operable, without executing a program instruction, to generate the first information from the second information. See column 7, lines 14-32, and note that second information (IP address) is used to generate first information through route table lookup.

32. Referring to claim 77, Inagaki, as modified, has taught the pipeline accelerator of claim 76 wherein the second information equals the first information. See column 7, lines 14-32. Basically, IP information (second information) is mapped to particular routine module

information (first information), which in turn identifies the destination of the processed packet. Because the IP and routine module information are mapped 1-to-1, they are equal. That is, both the IP and routine module information identify a destination.

33. Referring to claim 83, Inagaki, as modified, has taught a method as described in claim 43. Furthermore, claim 83 is rejected for the same reasons set forth in the rejection of 69.

34. Claims 13-14, 47-48, 68, 71-72, 74-75, 78-79, 82, and 84-85 are rejected under 35 U.S.C. 103(a) as being unpatentable over Inagaki in view of the examiner's taking of Official Notice and further in view of Bishop et al., U.S. Patent No. 4,914,653 (herein referred to as Bishop).

35. Referring to claim 13, Inagaki, has taught an accelerator comprising:

a) first and second memories. See 1, and note first memory 150-154 and 117 and second memory 119 and 124.

b) a hardwired circuit (Fig.1) coupled to the first and second memories and comprising:

- b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having first information specifying at least one destination hardwired processing element, to extract the raw data from the message, and to load the raw data into the first memory. See Fig. 1, Fig.3, and column 5, line 65, to column 6, line 35. Packet 134 of Fig.3 is received on the upper bus 1. The destination 163 in the header indicates which of the accelerators will be used to process the message data and also indicates which RAM_x section the data will be stored in.
- b2) hardwired processing elements including the specified at least one destination hardwired processing element and including at least one other processing element, the

specified at least one destination hardwired processing element operable to process data.
See Fig.1, component 9 and 11-14.

b3) an interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the at least one destination hardwired pipeline specified by the first information without using a virtual address, and load processed data from the hardwired pipeline into the second memory. See Fig.1 and column 6, line 61, to column 7, line 13. It should be noted that, although a virtual address is used to indicate a pipeline, the use of this virtual address occurs before the data is sent to the pipeline for processing. The virtual address is used to determining which RAM_x section (Fig.1) that the data is stored in. The section the data is stored in determines which pipeline the data will be sent to. However, the providing itself does not make further use of the virtual address.

Hence, it can be said that the data is providing without using a virtual address. The virtual address is used to store the data to a RAM_x; not to provide it from RAM_x.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having second information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source by providing the processed data and the second information to the external source via at least one same bus line. See Fig.1, column 7, lines 14-32, and column 8, lines 62-67.

b5) Inagaki has not explicitly taught that the circuit is operable to perform the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed

steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

b6) Inagaki has also not taught that at least some of the above components are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify at least some of Inagaki's components to be pipelined.

c) Inagaki has also not taught an input-data queue coupled to the input-data handler and the pipeline interface, wherein the input-data handler is operable to load into the input-data queue a pointer to a location of the raw data within the first memory and wherein the pipeline interface is operable to retrieve the raw data from the location using the pointer. However, Bishop has taught the concept of queuing pointers to input data in order to reduce latency during

transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to include a multi-packet input buffer and an input-data queue for holding a pointer to a packet (raw data) within the buffer and then using the pointer to locate the desired packet.

36. Referring to claim 14, claim 14 is largely rejected for the same reasons set forth in the rejection of claim 10. Furthermore, Inagaki has not taught an output data queue coupled to the output-data handler and the pipeline interface, wherein the pipeline interface is operable to load into the output-data queue a pointer to a location of the processed data within the second memory, and wherein the output-data handler is operable to retrieve the processed data from the location using the pointer. However, Bishop has taught the concept of queuing pointers to output data in order to reduce latency during transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to include an output data queue coupled to the output-data handler and the pipeline interface, wherein the pipeline interface is operable to load into the output-data queue a pointer to a location of the processed data within the second memory, and wherein the output-data handler is operable to retrieve the processed data from the location using the pointer.

37. Referring to claim 47, Inagaki, as modified, has taught a method as described in claim 44. Furthermore, the method of claim 47 is performed by the circuit of claim 13. Consequently, claim 47 is rejected for the same reasons set forth in the rejection of claim 13.

38. Referring to claim 48, claim 48 is largely rejected for the same reasons set forth in the rejection of claim 9. Furthermore, Inagaki has not taught loading into an output message queue a pointer to a location of the processed data within the second memory, and wherein retrieving the processed data comprises retrieving the processed data from the location using the pointer. However, Bishop has taught the concept of queuing pointers to output data in order to reduce latency during transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to perform loading into an output message queue a pointer to a location of the processed data within the second memory, and wherein retrieving the processed data comprises retrieving the processed data from the location using the pointer.

39. Referring to claim 68, Inagaki has taught an accelerator, comprising:

- a) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".
- b) a hardwired circuit coupled to the memory, including at least one processing accelerator (see Fig.1, at least one of components 9 and 11-14), and operable to:

- b1) receive a message that includes data and that includes a header having information uniquely identifying each of at least one pipeline. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Packet 134 of Fig.3 is received on the upper bus 1. The destination 163 in the header indicates which of the accelerators will be used to process the message data.

- b2) extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with the at least one

accelerator identified by the information. See Fig.1 and column 6, lines 29-35, and note that the data to be processed is stored in the appropriate buffer of buffers 150-154. The data is then retrieved from the buffer and processed by the corresponding accelerator of accelerators 9 and 11-14.

b3) provide the processed data to an external source. See Fig.1, column 6, lines 61-65, and column 8, lines 62-67. Note that after the data is processed, it is routed to an external location.

b4) extract from the header the information. See column 6, lines 9-13.

b5) Inagaki has not explicitly taught performing the claimed steps without executing a program instruction. However, it should also be noted that Inagaki makes not a single mention of executing instructions to perform the claimed steps. Hence, it is unclear as to whether Inagaki executes instructions to perform the claimed steps. Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the circuit performs the claimed steps without executing a program instruction, but instead in hardware alone. For example, as is known, an FPGA can be programmed to perform steps in hardware only and without instruction execution.

b6) Inagaki has also not taught that at least the processing accelerators are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow

serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki's processing accelerators to be pipelined.

b7) Inagaki has further not taught storing a pointer to the extracted data in a location associated with the pipeline corresponding to the destination. However, Bishop has taught the concept of queuing pointers to input data in order to reduce latency during transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to include a multi-packet input buffer and an input-data queue for holding a pointer to a packet (raw data) within the buffer and then using the pointer to locate the desired packet.

b8) Inagaki, as modified, has further taught providing the retrieved data to the pipeline in response to the stored pointer. Clearly, in view of b7 above, such a limitation is inherent.

40. Referring to claim 71, Inagaki has taught a pipeline accelerator, comprising:

a) a memory. See Fig.1 and note that any combination of components 150-154, 117, 119, and 124, may be considered "a memory".

b) a hardwired circuit coupled to the memory (see Fig.1), and operable to:

b1) receive data without receiving, with the data, information corresponding to a post-processing destination of the data. See Fig.1, Fig.3, and column 5, line 65, to column 6, line 17. Essentially, a packet is received on the upper bus 1 and ultimately processed.

The post-processing destination isn't added to the packet until after it is processed. See column 6, line 61, to column 7, line 32.

b2) process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.1 and note that the data is processed by at least one of processing units 9 and 11-14 before being stored in and ultimately retrieved from memory 119/124.

b3) generate a message header that includes first information indicating a destination of the processed data without using a virtual address, generating a message that includes the processed data and the header, and then providing the message to the external source. See Fig.1, column 7, lines 14-32, and column 8, lines 62-67. Note that the virtual MAC address is not used in this generation.

b4) Inagaki has not taught that at least some components are pipelined. However, Official Notice is taken that pipelined processing and its advantages are well known and accepted in the art. Specifically, pipelines provide an alternative to slow serial execution by dividing execution into stages and overlapping stages of different items being processed. This parallel form of execution is akin to an assembly line, which results in increased efficiency and throughput. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki's processing components to be pipelined.

b5) Inagaki has also not taught that claimed steps are performed without executing a program instruction. However, Official Notice is taken that hardware and software are logically equivalent and that anything performed by software can be performed solely by

hardware and vice-versa. The choice between hardware and software depends on desired speed, cost, and complexity. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki such that the claimed steps are performed without executing a program instruction, but instead in hardware alone.

b6) Inagaki has not taught storing a pointer to the processed data, storing in association with the pointer second information indicating the destination of the processed data, retrieving the processed data in response to the pointer, and generating the message header in response to the second information. However, Bishop has taught the concept of queuing pointers to output data in order to reduce latency during transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to store a pointer to the processed data, and retrieving the processed data in response to the pointer.

b7) Inagaki has further taught storing in association with the pointer second information indicating the destination of the processed data and generating the message header in response to the second information. See Fig.3 and column 7, lines 14-32. Note that the IP destination is again stored and used to generate the header.

41. Referring to claim 72, claim 72 is largely rejected for the same reasons set forth in the rejection of claim 71. Furthermore, Inagaki has not taught to store a pointer to the processed data in a location with the destination of the processed data, to retrieve the processed data in response to the pointer, and to generate the message header in response to the location. However, Bishop has taught the concept of queuing pointers to output data in order to reduce

latency during transmission. See column 2, line 62, to column 3, line 5. Therefore, in order efficiently transmit packets, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Inagaki to store a pointer to the processed data in a location with the destination of the processed data, to retrieve the processed data in response to the pointer, and to generate the message header in response to the location.

42. Referring to claims 74-75, claims 74-75 are rejected for reasons set forth in rejections of previous independent claims (most notably, but not limited to, 73 and 13).

43. Referring to claim 78, claim 78 is rejected for reasons set forth in rejections of previous independent claims (most notably, but not limited to, 76 and 14).

44. Referring to claim 79, claim 79 is rejected for reasons set forth in rejections of previous independent claims (most notably, but not limited to, 76 and 14).

45. Referring to claim 82, claim 82 is rejected for reasons set forth in rejections of previous independent claims (most notably, but not limited to, 48 and 14).

46. Referring to claim 84, the method of claim 84 is performed by the accelerator of claim 71. Consequently, claim 84 is rejected for the same reasons set forth in the rejection of claim 71.

47. Referring to claim 85, the method of claim 85 is performed by the accelerator of claim 72. Consequently, claim 85 is rejected for the same reasons set forth in the rejection of claim 72.

Allowable Subject Matter

48. Claims 66-67 and 80-81 are allowed.

Response to Arguments

49. Applicant's arguments filed on June 7, 2010, have been fully considered but they are not persuasive.

50. All arguments are non-persuasive for the reasons set forth in the rejections above. It should be noted that the examiner has allowed claims 66-67 and 80-81 since these are worded differently than the other independent claims. Also, regarding applicant's argument of the examiner's taking of official notice that it would be obvious to modify the accelerators of Inagaki to be hardware logic that processes data without executing software, the examiner asserts that Tanenbaum has already been cited in support of that position. See the PTO-892 mailed on July 20, 2009. Essentially, Tanenbaum states that hardware and software are logically equivalent. For instance, one could perform multiplication via software routine, or applicant could build a hardware multiplier that takes, as inputs, two operands to be multiplied, and outputs the multiplication result. Generally, a hardware-only implementation is faster than a software implementation because a software implementation requires software and hardware (as opposed to just hardware), and other known factors contribute to delays in software processing (dependencies, structural hazards, etc.). Hence, the examiner maintains that it is obvious to have the accelerators of Inagaki process data based on the data alone, and not based on an instruction.

51. Regarding the argument for claim 7, and other similar claims, that information corresponding to a post-processing destination is received with the data, the examiner disagrees based column 7, lines 14-32. The route table information, which is the only information being relied upon to be the information corresponding to a post-processing destination is not looked up until the data is received after processing and is ready to be routed somewhere else.

Conclusion

52. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

Aizono et al., U.S. Patent No. 6,282,578, has taught a distributed processing system with message handling and application initiation. See Figs.2, 15, and 19.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to DAVID J. HUISMAN whose telephone number is (571)272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/David J. Huisman/
Primary Examiner, Art Unit 2183